Bleeding segmentation of surgical images

Tian Zhao
Technical University Dresden
Faculty of Computer Science
Dresden, Germany

Abstract—Image segmentation is an important branch in the field of machine learning. Its main purpose is to distinguish the different areas in the picture by using different markings. To help the doctors during surgical operation, a method to identify the presence and location of blood is wanted. In this project, the segmentation will be done by using convolutional neural networks. The involved models are UNet, Multi-Output UNet, FCN_ResNet50 and DeepLabV3_ResNet50. The dice loss is used to describe the difference between ground truth and estimated segmentation. In the dataset, 70% of the images contain blood areas, while other images do not. Due to the inaccuracy of manual annotation, the validation loss of all models is about 0.26. But the estimated segmentations are good enough to describe the presence and location of blood. As final result, the UNet and Multi-Output UNet have both 0.269 as stabilized loss. For FCN_ResNet50 and DeepLabV3_ResNet50 are 0.266 and 0.252.

Index Terms—Computer Society, CNN, UNet, ResNet, Segmentation.

1 Introduction

THE researches on surgical assistance will make surgery safer and more efficient. Direction of current development is fully automated robot surgeons. The process of achieving this goal is divided into many steps and image segmentation is the first of these steps. Image segmentation is a computer vision task that marks specified areas based on image content. It divides the input image into multiple categories based on certain criteria, and will be regarded as input information for the subsequent steps of surgical assistance. As the goal of this project, the pixels should be separated in two categories: blood or background.

Image segmentation has a long history of development. As classical methods, there are already large mount of successful methods. For example thresholding [1], Clustering [2] and Histogram-based [3] methods. But after 2012, it has been significantly improved by using deep learning methods such as convolutional neural networks (CNN). The detection of blood in the picture will also be done by CNN.

This project is started with 9 surgery videos and as the first step, a dataset should be created. Frames from those videos will be picked up and labeled as blood or non-blood. During surgery, docters are moving carefully and slowly. Therefore, the difference between adjacent frames is sometimes very small, which leads to duplication of the training set. To avoid this problem, only one frame will be picked up in every 10 seconds. For each selected frame, a black-white mask should also be created as the ground truth for segmentation. Blood areas are labeled as white and background area as black. To reduce the size of dataset and the workload of labeling, blood areas will be marked with polygons and written as list of (x,y) position. The first contribution of this project is the script, which can read the polygon-informations from Excel data and create ground truth mask as JPEG file.

All ground truth masks are drawn using polygons, which causes the edges of the image to always be straight and chamfered. Due to the deep layers in the CNN model, the estimated blood area always has curved edges. It can be seen from this point that the difference between the ground truth and the estimated mask is inevitable. Furthermore, there is no clear definition of the blood range during the annotation process. The surgical images are manually annotated by the annotator and subjectively judged. The annotator may have different definitions of bleeding range at different times. If those ground truth are used in the evaluation process, compare to those well-knowned datasets, there will be a higher difference between the ground truth and the estimates. During the operation, the detected bleeding area will trigger certain response. From this point of view, a high precision is more important than recall.

In order to find a training model suitable for the surgical image database, multiple models are trained using the appropriate parameters. The models involved are UNet [4] and ResNet-based segmentation networks: FCN_ResNet [5] and DeepLabV3_ResNet [6]. After that, a Multi-Output UNet [7] is also trained with same parameters as UNet. Models are trained for 100 epochs. Based on confusion matrix and also properties from it, models will be evaluated and compared.

2 Methods

2.1 Dataset

Dataset creation is a very important part of machine learning. Some pre-process methods will improve the training result by solving specific problems. In order to evaluate the model fairly, the image is divided into five cross-validation groups, and the model will be trained five times respectively. To prevent overfitting, images will also be randomly crop and flipped. More details are provided in the following paragraphs.

2.1.1 Dataset prepairation

The Excel file contains blood area information for a total of 946 images and all of them will be placed in the dataset. Images without blood area will be randomly picked up from surgery videos. If the number of non-blood images is too small, the data set will be very different from the actual situation. Observation of the surgical video showed that the bleeding only occurred for a short period of time during the operation and soon got handled. If the number of non-blood images is too large, the training time will increase. As a compromise, the proportion of bleeding and non-bleeding pictures was 70% and 30%, respectively.

2.1.2 K-Ford validation

Dataset are split into five groups with some specific rules. For each validation, one of them will be treated as validation group and the summary of others as training set. Images from the same procedure often have high similarity. If the images are seperated ramdomly, there will be many similar pictures in the training set and evaluation set. This problem leads to a ”fake” validation, because some validation images are already learnd by the model. To avoid such situation, the videos are divided into five groups, ensuring that images from one recorded surgery are all contained in the same group. Before splitting, the number of annotated images in each video is counted. To ensure that each group has the same number of images, the videos are then divided based on the counted number. Using this method, the training and validation sets are always separate.

2.1.3 Random crop and flip

Overfitting is also a well-known problem within machine learning. The size of the data set cannot be endless, so it is impossible to fully reflect the real situation. Some features in the data set are related to training expectations, but there will always be irrelevant feature. If the estimation is strongly related to those irrelevant feature, the predictions for actual data will be poor. Such problem is called overfitting.

To prevent overfitting, some random linea transformations are used to break the irrelevant feature. The involved transformations are RandomResizedCrop() and RandomHorizontalFlip() from torchvision. By RandomResizedCrop, a smaller area from original image will be randomly selected then resized to 256×256. The scale parameter is settled in range 0.5 to 1. So the transformed image contains minimum 1/4 from the original one. After that, there is a 50% chance that the image will be flipped horizontally.

2.2 Evaluation

2.2.1 Confusion matrix

The confusion matrix is a situation analysis table that summarizes the prediction results of the classification model. According to the judgment of the two categories based on the real category and the classification model, the estimation results are summarized into the matrix form showed in Table 1.

		Real category	
		Positive	Negative
Estimation	Positive	TP	FP
	Negative	FN	TN

TABLE 1: Confusion matrix

The goal of training is to maximize the True Positive(TP), True Negativ(TN) and minimize False Positiv(FP), False Nagativ(FN). The prediction will not be perfect, so FP and FN will generally not be 0. In order to evaluate the training results, the importance of FP and FN must be analyzed. The number of FP represents the ”fake” bleeding pixels in prediction mask. As the opposite, FN represents that a bleeding pixel is ignored by the model. If the FP is too large, the surgeon will often be interrupted by a false bleeding alarm. If the FN is too large, although the model ignores some bleeding, the surgeon can still find it. It is not difficult to see that the optimization of FP has a higher priority than FN. This will serve as a basic rule for subsequent analysis of the evaluation method.

There are many evaluation methods which are extended from confusion matrix. The methods involved in this project are:

$$recall = \frac{TP}{TP + FN} \tag{1}$$

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$specificity = \frac{TN}{TN + FP} \tag{3}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$f_1 = \frac{2 \times precision \times recall}{precision + recall} \tag{5}$$

Following the above rules, the precision has top importance. The trained model will be evaluated in terms of precision and loss.

2.2.2 Loss function

Output of loss function presents the normalised difference between two images. If the imputed images are higI similar, the outputed loss will be close to zero. The dice loss [8] is defined as:

$$TP(I_m, I_e) = \sum_{x,y \in (0,255)} I_m(x,y) \times I_e(x,y) \tag{6}$$

$$Positiv(I) = \sum_{x,y \in (0,255)} I(x,y) \tag{7}$$

$$Dice\_Loss(I_m, I_e) = 1 - \frac{2 \times TP(I_m, I_e) + s}{Positiv(I_m) + Positiv(I_e) + s} \tag{8}$$

To avoid division by zero, set the static parameter s to 1. The number of blood pixels in image I will be calculated by equation (7). For the ground truth mask I_m, $Positiv(I_m)$ is a static value. If there are too many non-blood pixels marked as blood, the loss will increase by increasing $Positiv(I_e)$. If there are too many blood pixels marked as non-blood, the loss will increase by decreasing $TP(I_m, I_e)$. Due to these two characteristics, the loss function can correctly represent the difference between the two pictures.

The above loss function is used in all the following models.

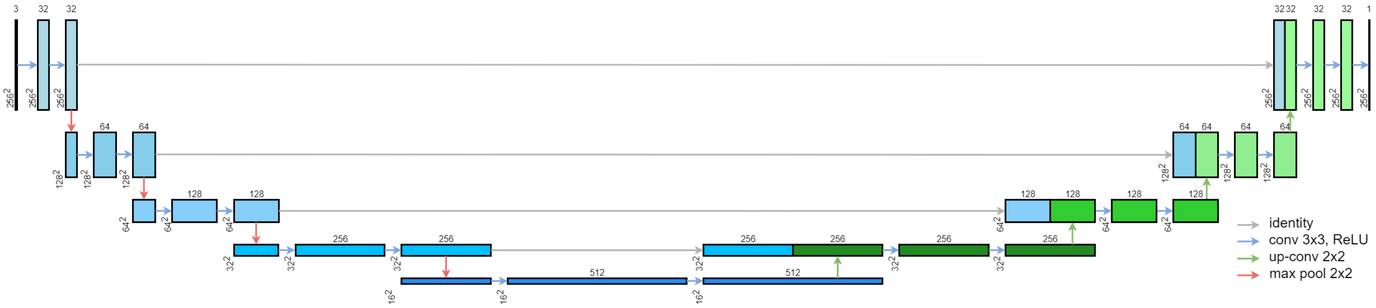

Fig. 1: Model architecture of UNet

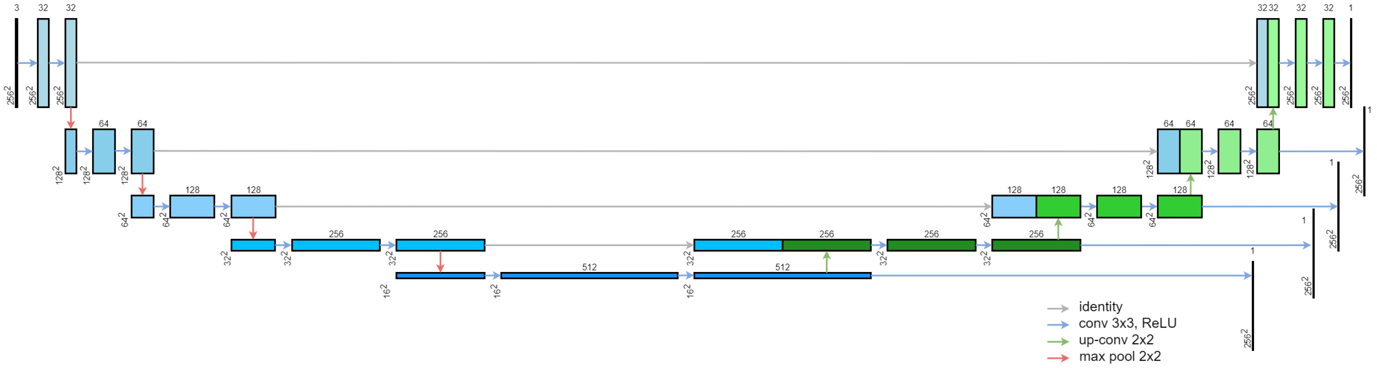

Fig. 2: Model architecture of Multi-Output UNet

2.3 Model Architecture

2.3.1 UNet

For image segmentation, especially medical image segmentation, UNet [4] is one of the most successful methods. This method was proposed at the 2015 MICCAI conference, and there are already more than 14000 references. The architecture of UNet is shown in Fig. 1. U-shaped structure and skip-connection are two very important features of UNet. Through convolution and max pooling layer, the image will be sampled 4 times, and the batch size is 2. Correspondingly, the high-level semantic feature will be restored to the resolution of the original picture through the up-sampling layer. Because of the U-shaped structure, the feature map incorporates more high-level features. The skip connection is used in each level of UNet, which ensures the low-level features in the feature map. In Fig. 1, each block represents the result of a convolution unit. The resolution of each block is displayed on the left, and the number of layers is displayed on the top. The image as input has 3 layers for three color channels, while the output has only one binary layer, representing blood or non-blood. The blue arrows indicate the convolution units, which is composed of a 2D convolution and a ReLU filter. The kernel size of 2D convolutions is 3×3. In order to find the suitable learning rate for UNet, this model has been trained many times with different learning rates. The decreasing loss curve for each learning rate is shown in Fig. 3.

The smaller the learning rate, the smaller the oscillation of the training curve, but it will take more time to reach the convergence state. To finish the training process with 100 epoch and 5-ford cross validation, it takes already sbout 24 hours. As a compromise, 0.0001 is the suitable learning rate for UNet and will be used for all models.

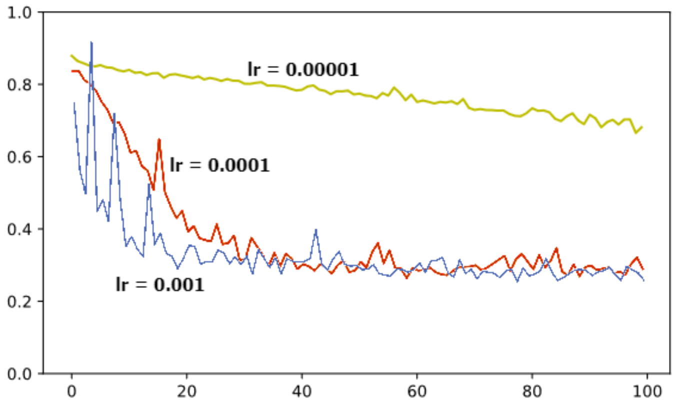

Fig. 3: Loss curve with different learning rate

2.3.2 Multi-Output UNet

Multi-Output UNet is a transformation of UNet. It will create five images from five different depths. Losses will be calculated separately using the same ground truth mask, and the weighted sum of all calculated losses is the final loss. In ordinary UNet, features from deeper layer will be compressed many times by convolution. And the output focus only on the result from lowest layer. By calculating the weighted sum of losses, Multi-Output UNet treats the convolution results of each depth fairly. The purpose is to make the estimation contain more general information and ignoring details. The architecture is shown in Fig. 2.

In implementation, the learning rate used is also 0.0001, and each cross-validation is trained 100 epochs. The same weight is given to all five images whensumming up losses, so the equation is defined as:

$$Loss_{sum}(I_m, \{I_i | i \in [0,4]\}) = \frac{\sum_{i \in [0,4]} loss(I_m, I_i)}{5} \qquad (9)$$

2.3.3 ResNet

The proposition of the deep residual network(ResNet) [9] is a milestone in the history of CNN images. It have emerged as a family of extremely deep architectures showing compelling accuracy and nice convergence behaviors. The involved models in this project are FCN_ResNet50 [5] and DeepLabV3_ResNet50 [6] from torchvision. Both models are used in not-pretrained model and with 2 as num_classes. The output of ResNet are the posibility that one pixel belongs to those classes. With the argmax function afterwords, the estimated class can be found. So the classes number in the output of ResNet should be 2 instead of 1.

There are also a special requirment, that the imput image should be normalized to [0.485, 0.456, 0.406] as mean value for three color channel and [0.229, 0.224, 0.225] as standard deviation. Those normalisation are done by using Normalize() function from torch vision.

3 Results

3.1 UNet

The evaluation result for UNet after 100 epochs are shown in TABLE 2 and the evaluation curve in Fig. 4 Some estimation examples are shown in Fig. 5.

In Fig. 5, green line represents ground truth area and yellow line shows estimation.

	loss	precision	recall	specificity	f_1
cross_val 1	0.1865	0.8739	0.7618	0.9896	0.8135
cross_val 2	0.3727	0.7479	0.5571	0.9848	0.6273
cross_val 3	0.3065	0.8558	0.5912	0.9956	0.6935
cross_val 4	0.1958	0.7421	0.8942	0.9672	0.8042
cross_val 5	0.2897	0.7781	0.6560	0.9918	0.7103
average	0.2702	0.7996	0.6920	0.9858	0.7298

TABLE 2: Evaluation of UNet

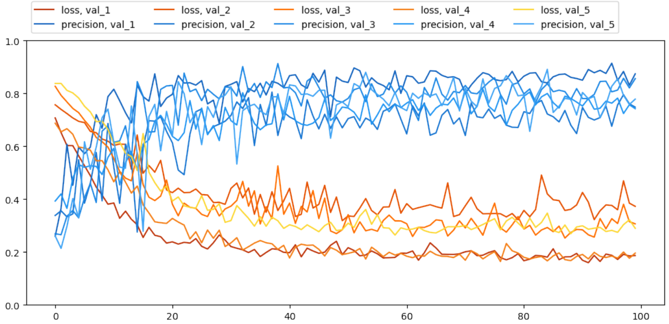

Fig. 4: Evaluation curve of UNet

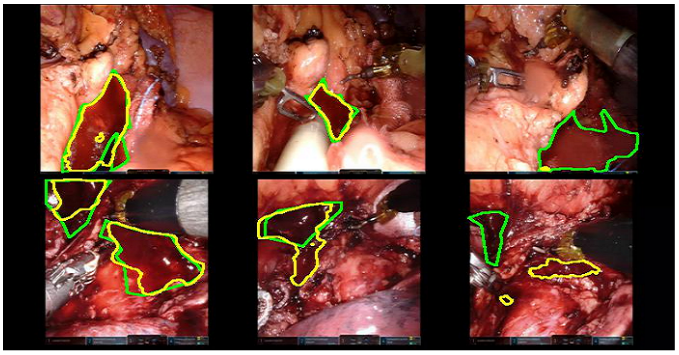

Fig. 5: Segmentation (as yellow) of UNet

3.2 Multi-Output UNet

The evaluation result for Multi-Output UNet after 100 epochs are shown in TABLE 3 and the evaluation curve in Fig. 6 Some estimation examples are shown in Fig. 7.

In Fig. 7, green line represents ground truth area and yellow line shows estimation.

	loss	precision	recall	specificity	f_1
cross_val 1	0.1806	0.8042	0.8409	0.9782	0.8194
cross_val 2	0.3637	0.7276	0.5727	0.9824	0.6363
cross_val 3	0.3284	0.8490	0.5691	0.9954	0.6716
cross_val 4	0.2144	0.7159	0.8763	0.9644	0.7856
cross_val 5	0.3107	0.8441	0.5859	0.9955	0.6892
average	0.2796	0.7882	0.6890	0.9832	0.7204

TABLE 3: Evaluation of Multi-Output UNet

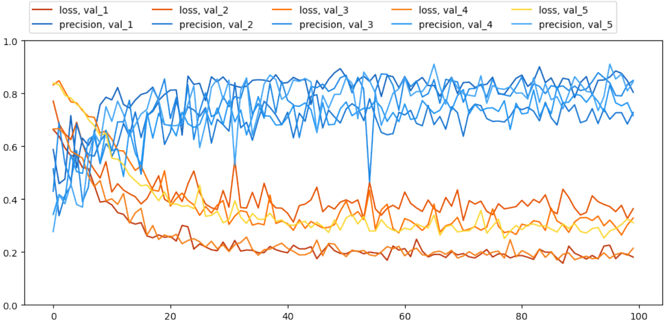

Fig. 6: Evaluation curve of Multi-Output UNet

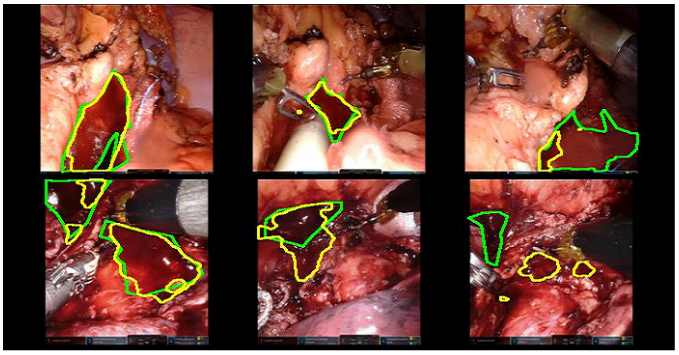

Fig. 7: Segmentation (as yellow) of Multi-Output UNet

3.3 ResNet-based segmentation networks

The evaluation result for ResNet after 100 epochs are shown in TABLE 4 and TABLE 5. In Fig. 8 and Fig. 10 are evaluation curves. The estimation examples are shown in Fig. 9 and Fig. 11.

	loss	precision	recall	specificity	f_1
cross_val 1	0.1582	0.9062	0.7875	0.9917	0.8418
cross_val 2	0.3586	0.7570	0.5697	0.9850	0.6414
cross_val 3	0.3576	0.8517	0.5684	0.9954	0.6424
cross_val 4	0.1856	0.7889	0.8818	0.9745	0.8144
cross_val 5	0.3386	0.7830	0.6201	0.9925	0.6614
average	0.2797	0.8173	0.6855	0.9878	0.7203

TABLE 4: Evaluation of FCN_ResNet50

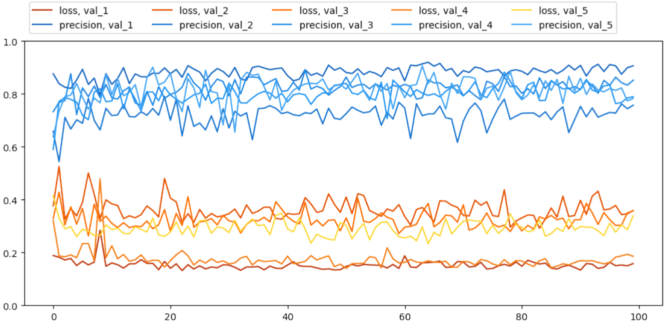

Fig. 8: Evaluation curve of FCN_ResNet50

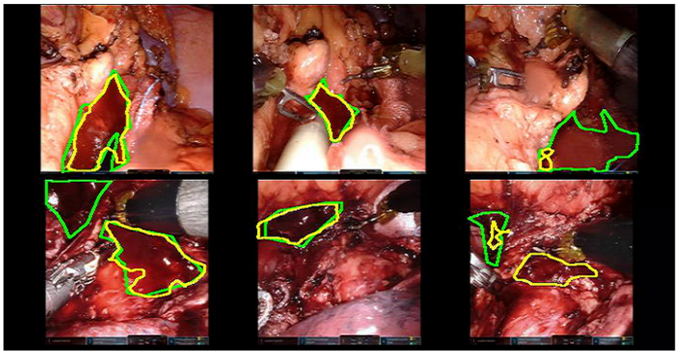

Fig. 9: Segmentation (as yellow) of FCN_ResNet50

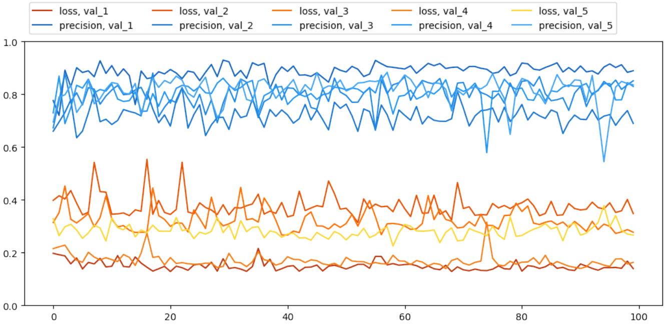

Fig. 10: Evaluation curve of DeepLabV3_ResNet50

	loss	precision	recall	specificity	f_1
cross_val 1	0.1395	0.8891	0.8406	0.9892	0.8605
cross_val 2	0.3478	0.6894	0.6597	0.9754	0.6522
cross_val 3	0.2772	0.8508	0.6857	0.9942	0.7228
cross_val 4	0.1630	0.8296	0.8807	0.9811	0.8370
cross_val 5	0.2662	0.8358	0.6541	0.9939	0.7338
average	0.2387	0.8189	0.7442	0.9868	0.7613

TABLE 5: Evaluation of DeepLabV3_ResNet50

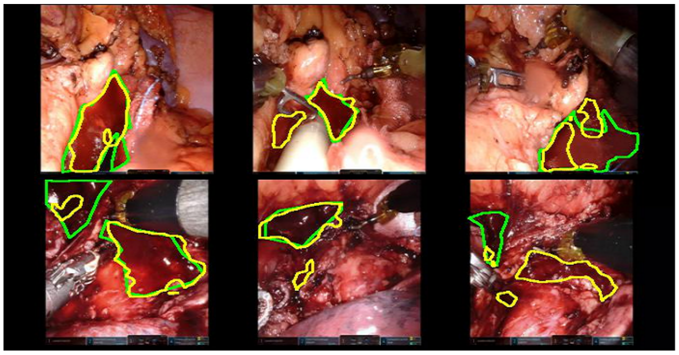

Fig. 11: Segmentation (as yellow) of DeepLabV3_ResNet50

3.4 Response time

The response time of segmentation using different models are shown in TABLE 6 below:

Response time(ms)	CPU (i7-6700k)	GPU (GeForce RTX 2070)
UNet	110	2.5
Multi-Output UNet	110	2.8
FCN_ResNet50	270	5.8
DeepLabV3_ResNet50	440	6.2

TABLE 6: Response time of image segmentation

4 Discussion

As can be seen from the tables, under a learning rate of 0.0001 and 100 epochs, the four models are not particularly different. DeepLab ResNet has the lowest average loss of 0.2387, while the average loss of other models is about 0.27. The prediction results of the ResNet model of FCN and Deeplab are 0.8173 and 0.8189. However, the precision of UNet and Multi-Output UNet are 0.7996 and 0.7882 respectively. Although it can be seen from the data that ResNet is better than UNet, the difference between the two is not significant. The training process of each model is shown in its line chart. After 20 epochs, the two ResNets have reached a convergent area but UNets need more than 40. One possible reason for this result is that ResNet has more convolutional layers. This can also be seen from the model parameter file saved at the end of training: the average size of the UNet parameter file is about 30MB, while ResNet need more than 120MB. Due to the difference in the number of model parameters, the response time when using the model to create predictions also varies greatly. Normal video has 60 frames per second, so in order to achieve real-time image recognition, the processing time of each frame needs to be less than 16.6 milliseconds.

From TABLE 6, it is easy to see that real-time segmentation requires the help of GPU. The experiments were done using ASUS GeForce RTX 2070 and the response time of segmentation with ResNet and UNet were 7 and 3.5 milliseconds, respectively. In addition to image segmentation, the program also needs to complete the preprocessing of the image, the conversion of the data format between the CPU and GPU and the final display in 16.6 milliseconds. From this point of view, UNet has the better chance to be real-time.

5 Conclusion

The first contribuition of this project is, that a new dataset of surgery images is created. There are four convolutional neural network models involved in this project: UNet, Multi-Output UNet, FCN_ResNet50 and DeepLabV3_ResNet50. Then the models are realized and all estimations are acceptable. The dice loss is used as loss function. The four models are also compared using loss, accuracy, recall and other parameters calculated from the confusion matrix. As a conclusion, the difference of accuracy between UNet and ResNet is very small. The response time of UNet is shorter than FCN_ResNet50 and DeepLabV3_ResNet50, which is important for real-time segmentation. The manual annotation of this dataset took a long time. A new method to annotate the blood area and the corresponding new loss function should be future work.

References

[1] Cheriet, Mohamed, Joseph N. Said, and Ching Y. Suen. ”A recursive thresholding technique for image segmentation.” IEEE transactions on image processing 7.6 (1998): 918-921.

[2] Coleman, Guy Barrett, and Harry C. Andrews. ”Image segmentation by clustering.” Proceedings of the IEEE 67.5 (1979): 773-785.

[3] Tobias, Orlando José, and Rui Seara. ”Image segmentation by histogram thresholding using fuzzy sets.” IEEE transactions on Image Processing 11.12 (2002): 1457-1465.

[4] Ronneberger O., Fischer P., Brox T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab N., Hornegger J., Wells W., Frangi A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science, vol 9351. Springer, Cham

[5] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. ”Fully convolutional networks for semantic segmentation.” Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[6] Chen, Liang-Chieh, et al. ”Rethinking atrous convolution for semantic image segmentation.” arXiv preprint arXiv:1706.05587 (2017).

[7] Sun, Tao, et al. ”Stacked U-Nets With Multi-Output for Road Extraction.” CVPR Workshops. 2018.

[8] Drozdzal, Michal, et al. ”The importance of skip connections in biomedical image segmentation.” Deep Learning and Data Labeling for Medical Applications. Springer, Cham, 2016. 179-187.

[9] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. ”Identity mappings in deep residual networks.” In European conference on computer vision, pp. 630-645. Springer, Cham, 2016.